Exploring Compute Cluster Dynamics Through Simulation

Mitch Richling

August 18, 2006

TEXAS INSTRUMENTS

Philosophical Ramblings

To Engineer

Engineering is a creative process, and engineers show a strong resemblance to other creative people. Painters & poets of technology if you will.

Philosophical Ramblings

To Engineer

Engineering is a creative process, and engineers show a strong resemblance to other creative people. Painters & poets of technology if you will.

IT's Responsibility

Enable that creative process without getting in the way. Engineers should be thinking about engineering, not about IT. They should be focusing on engineering tools, not how to get them to run in our grids.

Grid Vision

Basic Vision

Grid computing should be something treated as a utility on-tap in the Engineer's cubical. Something taken for granted most of the time. A seamless playground of compute. An endless sea of CPUs.

This must be done in a high performance and cost effective way.

Grid Vision

Basic Vision

Grid computing should be something treated as a utility on-tap in the Engineer's cubical. Something taken for granted most of the time. A seamless playground of compute. An endless sea of CPUs.

This must be done in a high performance and cost effective way.

Basic Implementation

Large, loosely managed general use grids are the work horse. They provide generic and simplified user interfaces while providing more sophisticated APIs for the development community. These large grids are shared across vast user communities.

Grid Vision

Disclaimer

This is not to say that we don't have small, special purpose grids. Sometimes this is the only way to meet special needs.

Even More Philosophical Ramblings...

Frame Of Mind

100% uptime? 0% job failure? Accept that perfection is an illusion, and that attempting to achieve it is simply an efficient way to throw away company revenue.

Even More Philosophical Ramblings...

Quantify "Good Enough"

Once we accept the futility of our pursuit of perfection, we open the door to quantifying just what is "good enough". The short answer: Just enough to maximize profit!

Even More Philosophical Ramblings...

New Questions

- Should I spend 20% of my data center funds on redundant power to avoid 1 day of down time, or should I get 20% more compute?
- Should I try to design single hop networks, or is it good enough that 90% of packets have only one hop?
- ...

The Big Idea

Thesis

Large, loosely managed general purpose compute grids (grid on-tap at the cubical) may be successfully managed much like a casino.

The Big Idea

Thesis

Large, loosely managed general purpose compute grids (grid on-tap at the cubical) may be successfully managed much like a casino.

Note About Applicability

This is not to say that all compute clusters can be managed in this way, just ones that meet our criteria – discussed later. Casino-style techniques may also be applied to such specific environments; however, the modeling required is much more detailed.

Successful Utility Grids & Casinos

Successful Casino

1. Large consumer population
2. Event probabilities favor the house

Successful Utility Grid

1. Large population
2. Event probabilities in favor of user success

Mathematical Requirements: "*Large Population*"

- Diversity is essential for mathematical reasons! Diversity of user population, work patterns, applications, hardware types.
- Large population of not just users, but applications, work patterns, hardware types, and job counts.

Summary Statistics In Action

Some Summary Statistics

Suppose we have a large job population consisting of jobs that all run about the same amount of time. Each job has one CPU dedicated to it on a 1 CPU compute server. The farm generally has no free job slots, and the CPU utilization runs at 50% most of the time.

Summary Statistics In Action

Some Summary Statistics

Suppose we have a large job population consisting of jobs that all run about the same amount of time. Each job has one CPU dedicated to it on a 1 CPU compute server. The farm generally has no free job slots, and the CPU utilization runs at 50% most of the time.

Reasonable Conclusion

With the given, extremely limited information, it is reasonable to assume that we can run two jobs per CPU in such an environment.

Summary Statistics In Action

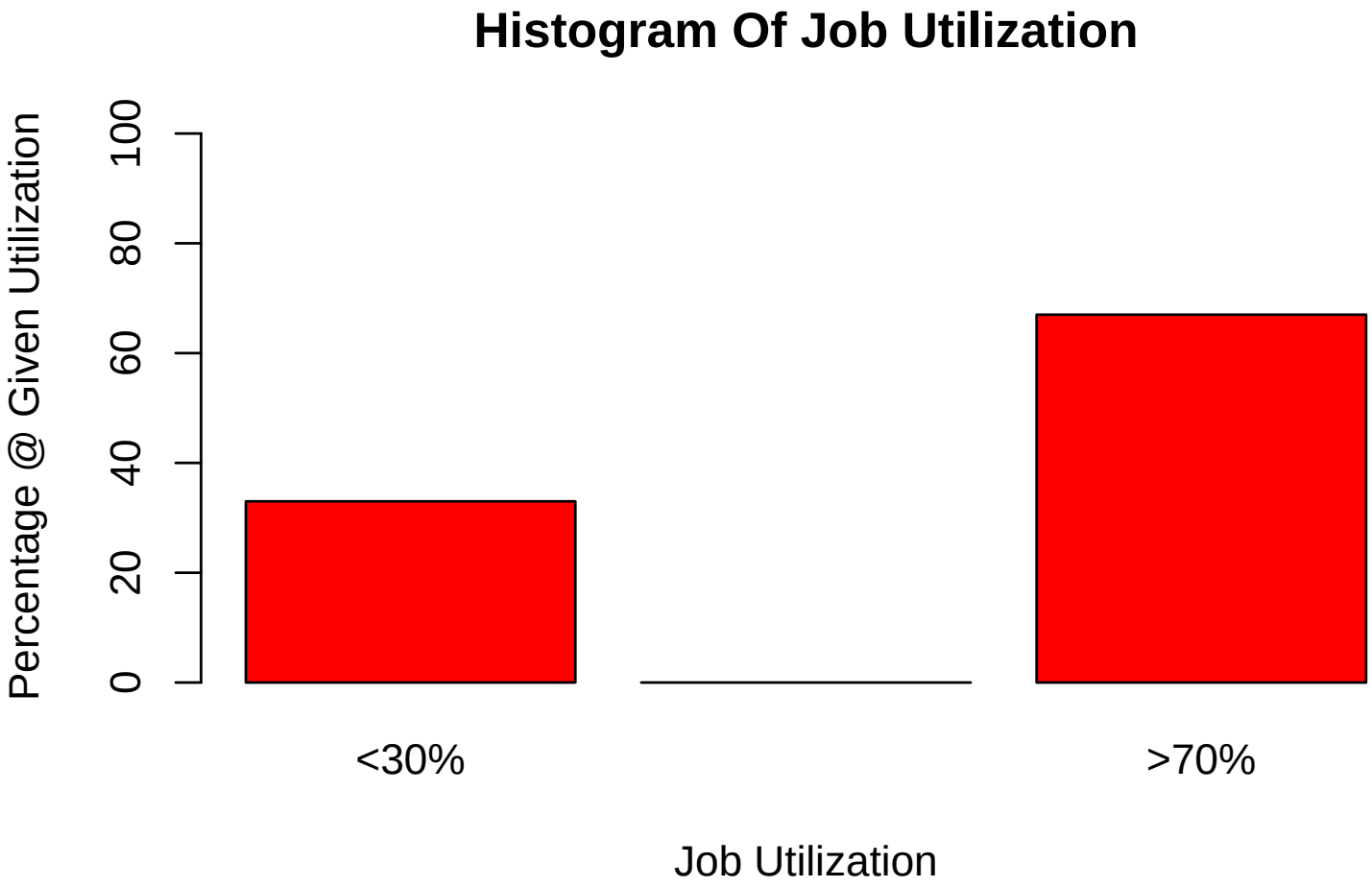
Some Summary Statistics

Suppose we have a large job population consisting of jobs that all run about the same amount of time. Each job has one CPU dedicated to it on a 1 CPU compute server. The farm generally has no free job slots, and the CPU utilization runs at 50% most of the time.

WRONG Conclusion

With the given, extremely limited information, it is reasonable to assume that we can run two jobs per CPU in such an environment.

The Histogram: Thinking Probabilistically



The Histogram: Thinking Probabilistically

Histogram Information Changes Our Perspective

Histograms force us to begin to thinking probabilistically, and that leads to experimental designs to answer questions. With this new information we can begin to consider the more combinatorial nature of the problem, and view the world from the perspective of game theory.

A First Experiment

What happens when one randomly combines two jobs from the previous distribution onto a single CPU host?

Thought Experiment

The Experiment

Randomly combine two jobs on a CPU. 67% of jobs with high utilization, and the rest with 0%.

The possibilities

Job 1	Job 2	Probability	Cost
H	H	44%	2x
H	L	22%	1x
L	H	22%	1x
L	L	11%	2x

Thought Experiment

The Experiment

Randomly combine two jobs on a CPU. 67% of jobs with high utilization, and the rest with 0%.

The cost

We see that the average job slow down is around 1.5x!

$$(2)(.55) \cdot (1)(0.45) = 1.55$$

Thought Experiment

The Experiment

Randomly combine two jobs on a CPU. 67% of jobs with high utilization, and the rest with 0%.

The cost in the general case

If p_h is the percentage of "high utilization" jobs in the cluster, then one has an average slowdown of:

$$2\left(p_h - \frac{1}{2}\right)^2 + \frac{3}{2}$$

That is a quadratic function. Things get bad very quickly indeed!

Thought Experiment

The Experiment

Randomly combine two jobs on a CPU. 67% of jobs with high utilization, and the rest with 0%.

The new conclusion

Given the histogram data and the result of this experiment, it is quite unlikely that one would make the decision to run two jobs per CPU!

The Evil Specter Of Intractability

Observation

While thought experiments and probability calculations are entertaining; they quickly become mathematically intractable as the scenarios under study become even slightly complex.

The Evil Specter Of Intractability

Observation

While thought experiments and probability calculations are entertaining; they quickly become mathematically intractable as the scenarios under study become even slightly complex.

The painful alternative: Real Experiments

Performing experiments with real hardware is a common alternative solution; however, it is less than desirable for obvious reasons.

Simulation Is An Answer!

Modeling

A *model* is a computer program that mimics the behavior of a real system. For example, we might model a CPU with a program that takes two job descriptions (wall time and CPU time) and computes what might happen if both jobs ran on a CPU.

Simulation

A *simulation* is a program that drives a *model* with random data based upon histograms or with previously recorded data from a real system. For example, we might throw random job characteristics (wall time and CPU time) at the CPU model above.

Simulation Is An Answer!

- Transform intractable probability computations into simple, brute force calculations!
- Utilize existing experience with physical experiments
  - Experimental design
  - Interpretation of results
- Cost significantly less than doing a real evaluation
- Make use of cluster resources. You have a grid. Use it!

Simulation: The modeler's favorite tool

Simulation has become the de facto technique of choice in many mathematical fields with difficult probability problems. Mathematical queuing theory and game theory are prime examples of fields peppered with problems that are only approachable via probabilistic simulation techniques.

Simulation Is An Answer!

Simulation Utopia

Simulation combines our already existing skills for doing physical experiments with the cost savings of probabilistic thought experiments. In addition, all of the "hard math" required for probabilistic thought experiments replaced by extra compute capacity in our grids.

Simulation Recipe

- A clearly defined questions
- An experiment to answer the questions
  - Well defined and measurable variables
  - Clear connection between measurable variables and answers
- Histograms describing experimental parameters
  - The environment (hosts, users, jobs, etc...)
  - The experiment (behavior)
- Some software
  - A program modeling the environment under test
  - A simulation program to drive the model with random data
  - Good random number generator to provide random data
- A little bit of math (to set things up and analyze results)
- Compute capacity to run simulation

Obtaining Histograms

General Advice

- Identify the "defining" qualities of your experiment.
- Do not confine yourself to uniform bucket sizes!
  - More resolution, smaller buckets, where data is highly variable.
  - Less resolution, bigger buckets, where data is less variable.

Advanced Advice

- Normalize histogram data into manageable ranges. For example, express RAM in units of GB instead of bytes.
- Synthetic histograms derived from other histograms can save considerable compute time over directly using the source histograms.
- Represent multidimensional random variables directly as multidimensional histograms instead of using several independent histograms.

Obtaining Histograms

Histograms For Batch Data

I use a perl script that parses the `lsb.acct` files along 20 different variables to produce a 20 dimensional histogram. The script automatically adjusts the bucket sizes so that the data is evenly distributed over the buckets. Generally, I end up with approximately 40K buckets. Bucket count doesn't vary appreciably over time ranges from one day to one year. It also automatically scales the data into reasonable units.

Returning to our example....

Making things more complex

- Let's run 3 jobs on 2 CPU hosts during working hours.
- Realistically compute run times based upon native job utilization. i.e. a 70% CPU job and a 40% CPU job only slow down about 10%.
- Assume best case behavior for low utilization jobs.
- Answer a new question: What is the average slowdown for a typical second of compute?

Returning to our example....

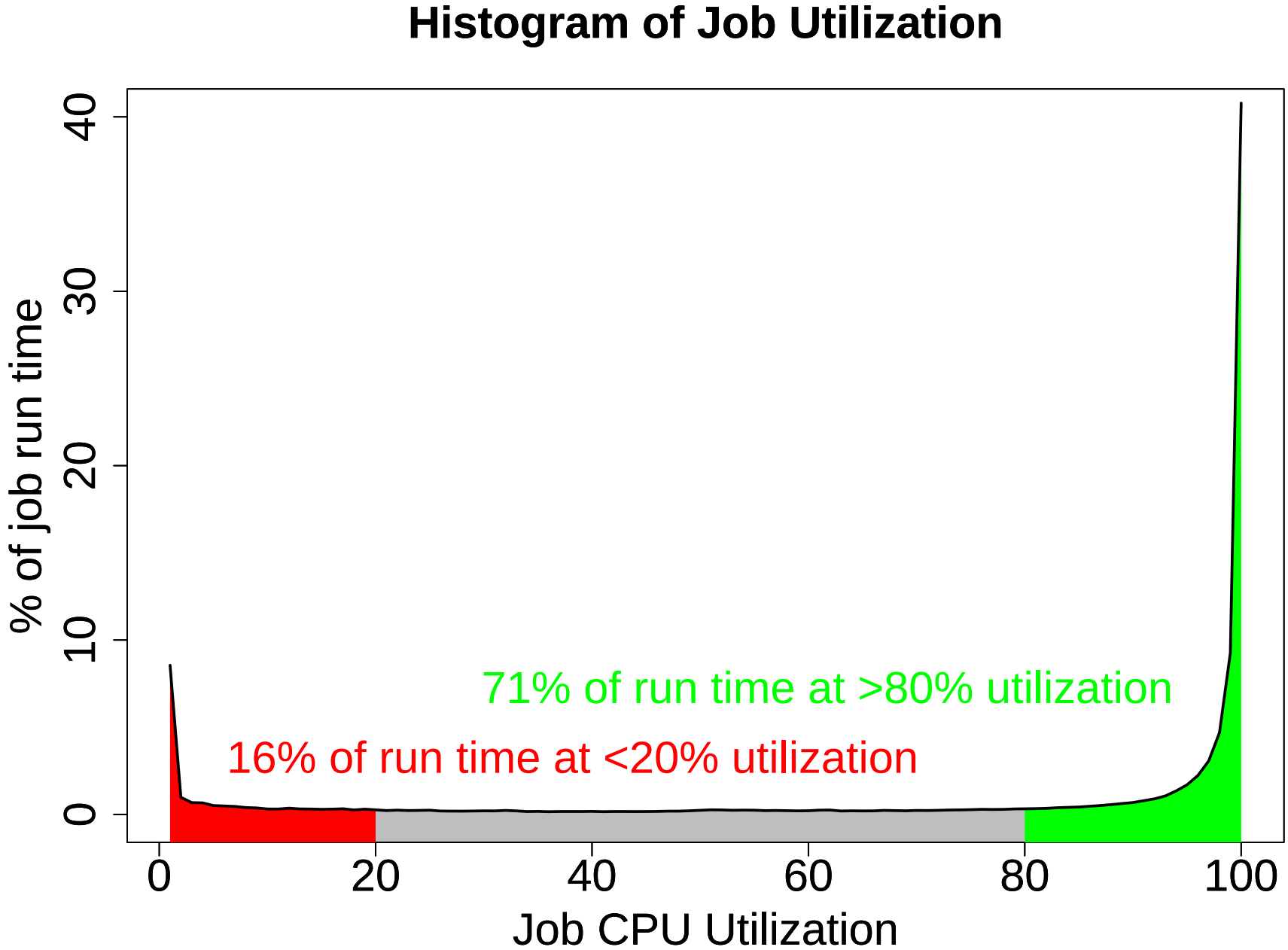
Histograms we will need

- Job CPU Time Consumed
- Wall Clock Time Consumed
- Time of day
- Day of the week (over a time with no holidays)

Returning to our example....

Explaining the results to management

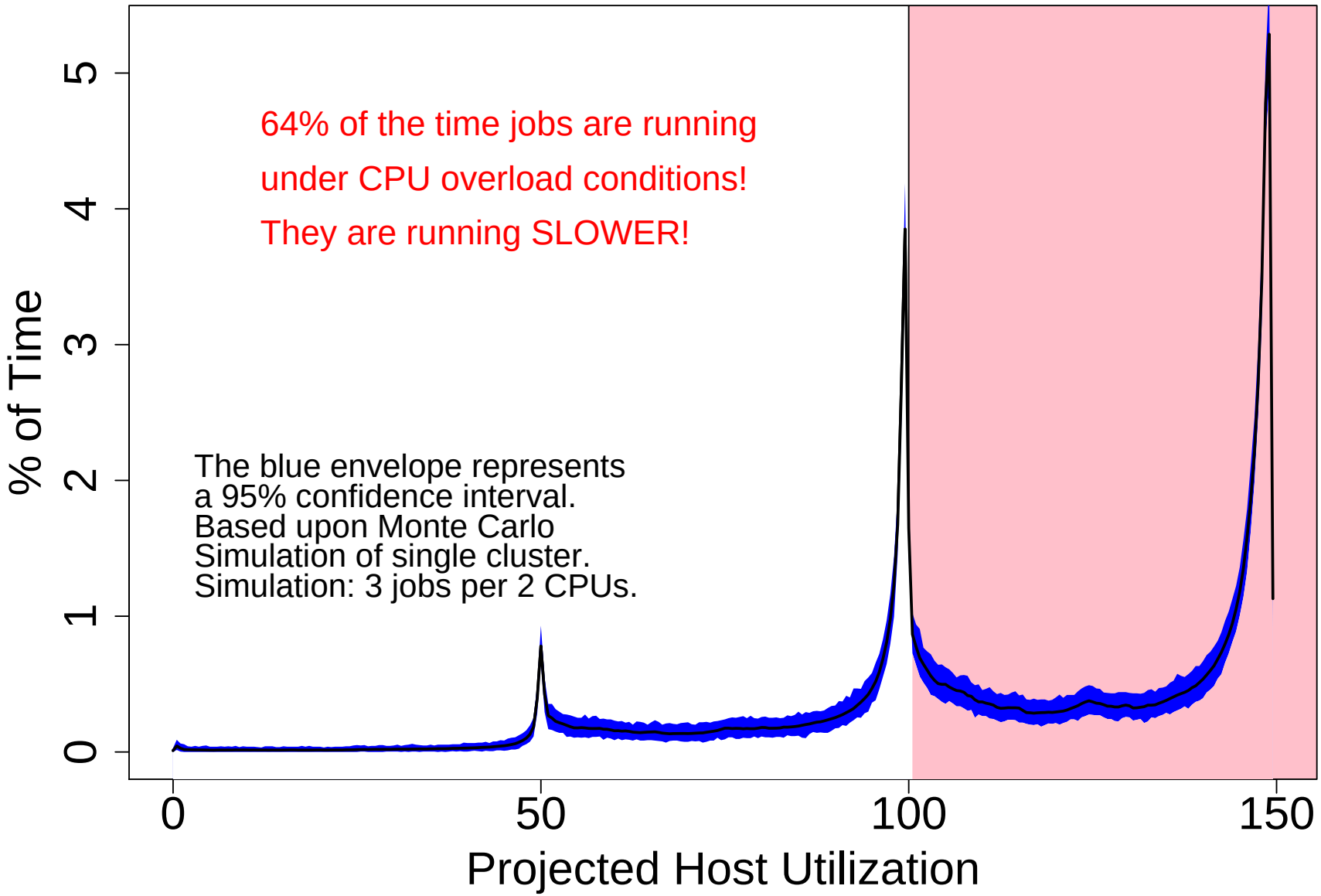
Don't ever display your input histograms and the output results and expect people to understand and accept them. Instead, combine the the important histograms together into a graph that explains the kernel of the behavior. Once you have a reasonable way to explain the nature of the results, only then will people truly believe them.



Graph By: Mitch Richling (2006−05−09)

Data is for 2006Q1 from a single TI cluster. Only jobs run on 2CPU linux boxen

Host Utilization Histogram (1.5 Slots/CPU)



Graph By: Mitch Richling (2006−05−09)

Data is for 2006Q1 from single TI cluster. Jobs run on 2CPU linux boxen

Various LSF Applications

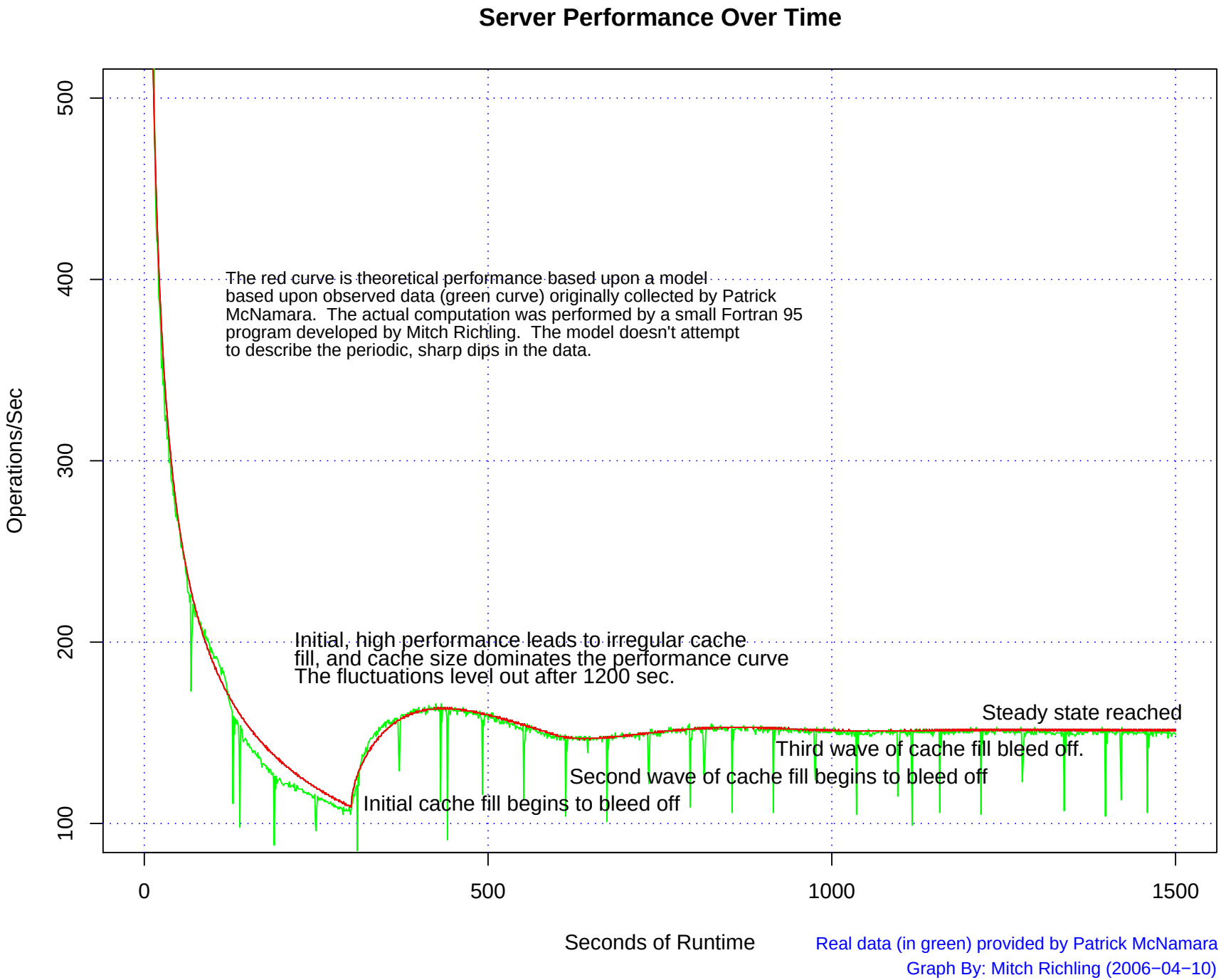
Past Projects

- Optimize slot/CPU configuration in LSF for maximum throughput.
- Determine correct chunk size for short queues.
- Characterize wait time in GUI queues.

Non-LSF Application: Characterize ClearCase Performance

The Problem

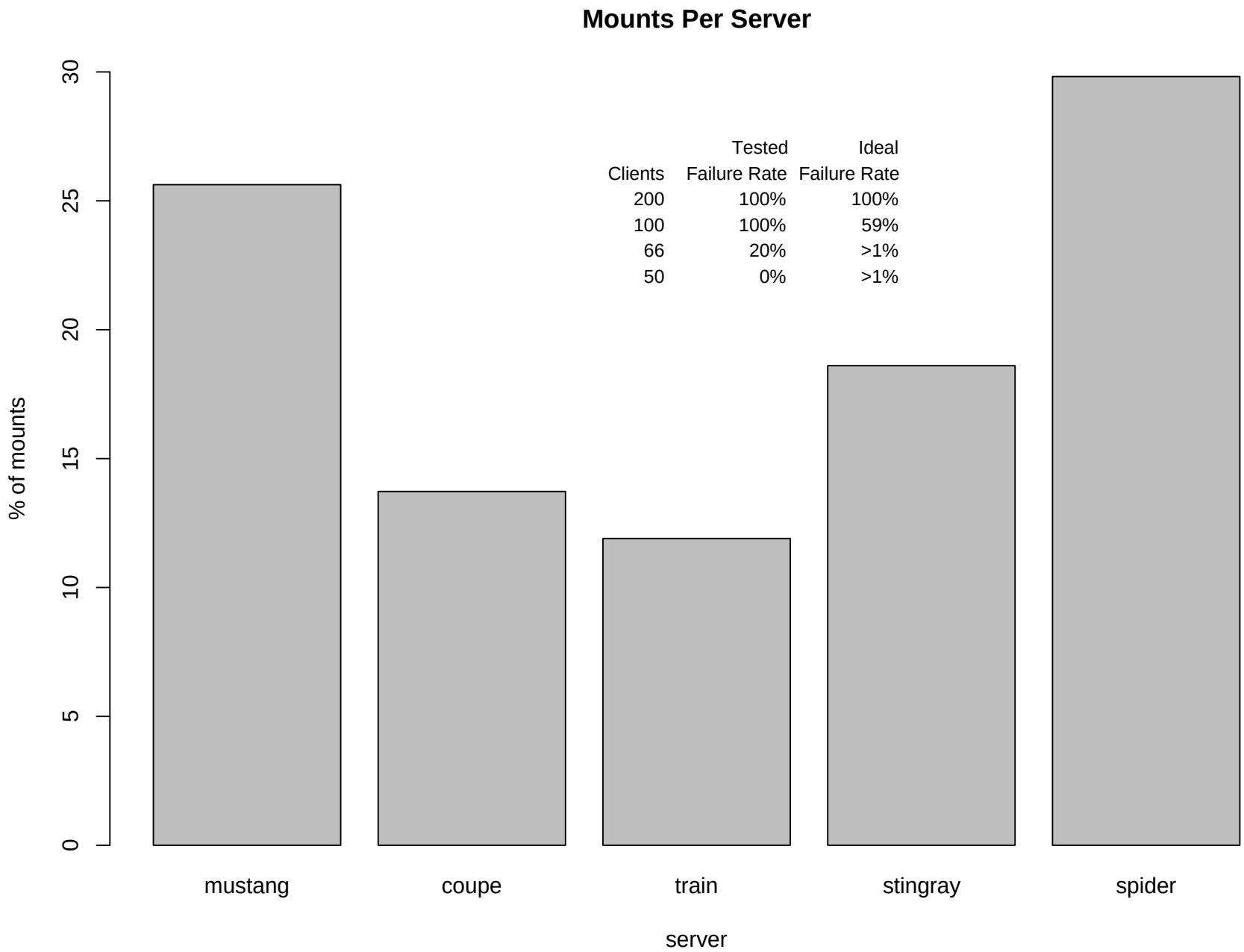
The task was to model a particular ClearCase daemon to determine the steady state performance. This would then tell us how many such servers we would need to deploy in parallel to meet the demand.


Server Performance Over Time
Operations/Sec
100
200
300
400
500
0
500
1000
1500
Seconds of Runtime
The red curve is theoretical performance based upon a model based upon observed data (green curve) originally collected by Patrick McNamara. The actual computation was performed by a small Fortran 95 program developed by Mitch Richling. The model doesn't attempt to describe the periodic, sharp dips in the data.
Initial, high performance leads to irregular cache fill, and cache size dominates the performance curve The fluctuations level out after 1200 sec.
Steady state reached
Third wave of cache fill bleed off.
Second wave of cache fill begins to bleed off
Initial cache fill begins to bleed off
Real data (in green) provided by Patrick McNamara
Graph By: Mitch Richling (2006−04−10)

Non-LSF Application: File Server Load Balancing

The Problem

Use the RHELv4 automounter to randomly select one of several replicated file servers. In this way the same data may be made available at higher bandwidth. The application will crash if any one file server has more than 20 mounts.


Mounts Per Server
% of mounts
0
5
10
15
20
25
30
mustang
coupe
train
stingray
spider
server
Clients
Tested Failure Rate
Ideal Failure Rate
200 100% 100%
100 100% 59%
66 20% >1%
50 0% >1%
Graph By: Mitch Richling (2006−04−21)
Servers listed in order of occurrence in mount map
Data from tests on 2006−04−21

Summary: Run Your Grid Like A Casino

- "Just enough to maximize profit" is the answer to the question "How much" in IT.
- Manage big grids like a casino. i.e. quantify "Just Enough" via probability theory and likely outcomes!
- Simulation is a viable tool for the weary IT admin.
  - Provides answers to hard probability questions.
  - Avoids costly physical evaluations and testing.
- Histograms are fundamentally important in the analysis of grid derived data even if simulation is not the final goal

Thank You!

TEXAS INSTRUMENTS

Support Material Follows

Final Remarks: Histograms In Cluster Measurement

Observations: Cluster Data

- Cluster data often is "non-normal" in mathematical terms. This means that most of the techniques from "Statistics 101" simply don't work.
- In environments with power users or power teams, often the data is not only non-normal, but highly polar with histograms showing two major peaks at the data extremes.
- When data is non-polar, it often shows multiple major peaks.
- Summary statistics often fail; however, combinatorial statistics are often better – mean vs. median for example.

Final Remarks: Histograms In Cluster Measurement

If you take away only one thing from this presentation:

Always consider histogram data when preforming any data analysis related to grid computing. Never depend solely on summary data.

Tools

- R is a free implementation of the S programming language, and has grown into a full blown statistical analysis package. R has wonderful graphics capabilities, and all graphics for this presentation were done with R.
- Breve is a 3D simulation environment than can be adapted to IT problems – with cool 3D graphics too!
- Mathematica from Wolfram Research is a general mathematical computing environment frequently used for modeling. I would suggest it over the common alternatives because it has an established IT modeling community.

Random Numbers

- Quality random numbers are essential for quality results from any simulation. Many packages are available:
  - `ranlib` is quality library available in both FORTRAN 77 and C versions.
  - The GNU Scientific Library also has quality random number generators – under a GPL license.
  - PRNG is a very good parallel random number generator.
  - The boost C++ library also has a very capable generator.
  - Finally, cryptographic sources like OpenSSL are good sources of random numbers.
- Example random number source code is available at:
  `http://homepage.mac.com/richmit/mitch/SITES/exampleCode/random.html`

General Programming Languages For Simulation

- C/C++ are common languages known in IT shops, and they both provide reasonable access to mathematical libraries (even in Fortran).
- Perl and Ruby are ideal languages for text processing – extracting data from log files for example. In addition to providing data to simulation programs, they can be used themselves for simulation implementation; however, performance can be an issue.
- Fortran 95 is an updated version of Fortran offering modern language features (OOP, pointers, etc..) combined with modern vector notation like that found in Matlab. It provides easy access to access the incredibly rich collection of mathematical libraries written in older Fortran.

Book Recommendation: Simulation

Introduction to Probability Models
By Sheldon Ross

This very popular book is commonly seen on the shelf even at non-technical book stores. It covers a wide range of topics from probability theory and Markov chains to simulation. It covers none of them very deeply. This book must be owned simply so that one can communicate with the many lay people who only know what can be found in this one volume.

Book Recommendation: Random Numbers

Random Number Generation and Monte Carlo Methods
By James E. Gentle

Gentle's book is a comprehensive and modern overview of random number generation techniques. Most of the important generators are covered in detail – algorithms, quality measures, and use patterns. The writing is clear, and the exercises are well motivated. The bibliography is extensive, and worth the price of the book by itself.

Book Recommendation: Statistics

Fundamentals of Modern Statistical Methods:
Substantially Improving Power and Accuracy
By Rand R. Wilcox

This book provides a whirlwind tour of the fundamental problems inherent in traditional statistical methods and some modern alternatives to get around them. This work is not mathematically deep or very precise, but it is a wonderful introduction to the subject. This is the book that I recommend to most people who need a basic understanding of why the statistics they had to learn in school simply don't work.

Book Recommendation: Computer Performance

Measuring Computer Performance: A practitioner's guide
By David J. Lilja

This book is a great read for non-expert statistical practitioners and IT personnel interested in IT performance analysis. It is a very focused introduction to the theory of measurement, experimental design, performance analysis, queuing theory, data analysis, and statistics as related to computer performance measurement. While not targeted for statisticians, mathematicians, or mathematical computer scientists, such readers may well find a few gems in this work.

Book Recommendation: Queueing Theory

Queueing Networks and Markov Chains
By Bolch, Greiner, Meer, & Trivedi

This book is a comprehensive introduction to the field from the most basic mathematics required for queueing theory to relatively advanced topics. It's well organized and written in a clear style. This is one of the only truly comprehensive introductions to queueing theory available today.